

第一部分 简 介

第1章 确定自己的正常工作时间需求

首先，请读者想象一下：您是一个网上书店的营销人员。每天，都有来自全球各地的用户登录到您的数据库中，查找自己所需要的书籍并购买这些书籍。即使是在最不忙碌的深夜，至少也有100个用户在访问您的数据库。因此，在任何时候将数据库系统关闭都是不可取的，而且任何影响性能的瓶颈也是不可接受的。这时，您就要求数据库系统必须全天候地高速运行。那么，面对下面的问题，该怎么办呢？什么时候执行数据库备份动作？什么时候进行数据库系统的碎片整理动作？什么时候执行数据库系统所必需的各种维护动作？

欢迎进入24×7的世界！术语“24×7”是在IT（Information Technology，信息技术）领域中，特别是在计算机系统以及数据库操作中频繁使用的一个术语，用来说明某种资源（例如，数据和系统与计算机系统）的连续可用性。也就是说，如果某个数据库/系统对于用户每天24小时、每周7天都是可用的，就把这个系统称为运行于“24×7”方式。

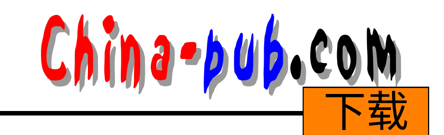
大多数公司对于术语“24×7”都非常了解，对于系统停工也不陌生，并且充分地认识到数据库可用性的重要性。但是，必须花费很高的代价才能保证提高系统运行的正常工作时间。即使是执行普通的数据库管理任务，都需要进行预先的计划、艰苦的工作等等。

很早以前，许多公司的操作管理员就要求有一个能够支持24×7正常工作时间的系统。直到最近，这些操作管理员才发现将自己的数据库进行升级并且使它运行在24×7方式下已经迫在眉睫。也许，DBA (Data Base Administrator, 数据库管理员) 会说，“这好办，可以使用大型机来解决问题！”。但是，在一个运行Oracle数据库系统的中、大型机器上也同样会产生各种错误。

现在，这种情况发生了变化：操作管理员仍然坚持希望自己的数据库系统能够连续地保持有效的工作；令人惊奇的是，DBA再也不会认为这样做毫无疑义而耸着肩走开。随着Oracle8与Oracle8i的发布，人们的注意力进一步集中到系统的可用性上。此外，随着电子商务与大规模全球市场的开发，DBA即使是在需要进行冷备份时，也不能随便地将数据库系统关闭。

在理想的情况下（不存在数据库崩溃、磁盘整理、外界环境灾难等等），即使是没有附加功能（例如备份）的Oracle7或者Oracle8也可以提供100%的系统可用性。但是，在真实世界中，如果没有提供其他附加的手段，想要保证系统工作在24×7方式下是不现实的。24×7方式要求用户对于系统的全部运行环境非常了解，能够预测到导致系统停工的各种可能情况，并且能够在不损失系统可用性的情况下对可能发生的意外作好处理准备。

Oracle7和Oracle8都支持数据库高可用性的许多功能。虽然，获取数据库系统99.99%的可靠性需要有非常巨大的额外开销（更别说是获取100%的可靠性了），但用户可以在不提供任何其他附加手段的情况下使用Oracle7或者Oracle8获取系统90%的可靠性。然而，您能够容



忍自己所提供的服务只有90%的可靠性吗？所谓90%的可靠性，也就是意味着必须容忍每天有2.4小时的系统停工时间、每10天就有1天的系统停工时间、每年有36.5天的系统停工时间。乍一看，似乎这些并不是任何需要有高数据库可用性的公司所需要的。但是，是不是您的公司就真的需要更高的系统可用性呢？本章将帮助读者确定自己数据库系统的可用性需求。

本章将讨论以下技巧与技术：

- 分析是否需要24×7正常工作时间的系统可用性
- 理解公司"数据库"的组成部分
- 分析什么部分容易导致错误以及错误出现的症状
- 熟悉构建24×7系统的共同目标
- 利用服务级协议管理意外事件
- 只有90%的系统可用性将节省90%的系统开销
- 理解系统维护操作对于系统可用性的影响
- 24×7正常工作时间对操作管理提出更高的要求
- 在关键性的数据库访问时间段创建"24×7小组"
- 维护一个关于数据库访问用户的可扩充列表，以便在危机时通知他们
- 在数据库系统停工之前，预先通知顾客/终端用户

1.1 理解24×7对于公司的重大作用

本节将列出24×7的一般内涵，然后说明系统中容易导致停工的各种组件。

1.1.1 分析是否需要24×7正常工作时间的系统可用性

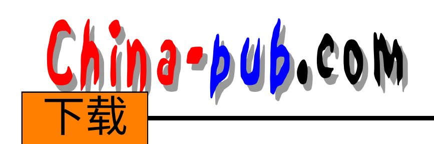
在确定是否需要24×7正常工作时间系统可用性之前，首先必须明确公司对于系统可用性的需求。在此，笔者将描述自己在两个客户站点的亲身经历。

三年前，我访问了一个位于芝加哥的公司客户站点（不妨把它称为公司A），这个公司主要从事旅游与售票服务工作。它有分布在北美洲与欧洲的上百个旅游代理，它们都需要访问位于公司总部的数据库系统，但这个公司中数据库系统运行并不稳定。笔者经过数小时运行utlbstat/utlestat，然后对数据库警告日志进行深入的分析之后，最终发现这个公司的数据库系统需要进行大规模的性能优化。最后，列出了10个能够对这个数据库进行优化的建议。这些建议都非常容易实现(至少本人认为如此)。

在笔者将自己列出的这些建议交给公司的DBA之后，他说自己也已经注意到了这些问题。那么，为什么不执行相应的动作来改善这种情况呢？因此，笔者决定绕开DBA而直接将自己的详细建议报告发送给这个公司的CIO(首席顾问)。

这些建议包括：

- 改变某些init.ora参数，重启数据库系统。
- 数据库系统被关闭之后，重新启动UNIX（因为内存已经被严重地分块）。
- 重新构建DATA与INDEX表空间（因为它们同样被严重地分块）。
- 在主表（包含有16兆行）上删除并重建四个索引。所有的这些索引都是"平面型的"，从而导致大规模的扫描动作（关于这个问题的详细原因，请参见第6章）。
- 任何时候都不要进行热备份（在此之前，即使是在用户访问的高峰期，他们也要每天花



费14个小时进行连续的热备份动作)。

看过这些建议之后,CIO问我是否执行所提出的这些建议需要将数据库系统进行重启,在得到我肯定的答复之后,我就被炒鱿鱼了。

那么为什么会这样呢?也许,高性能并不是这个公司的唯一需求,它们的需求是 24×7的数据库运行时间以及良好的性能。这是因为旅游代理可能会在一天中的任何时候访问数据库并且进行订票。哪怕是只有一个小时的数据库关闭时间,公司也有可能会损失不小的利益,因此必须保证系统具有24×7的可用性。

一段时间之后,笔者又有机会访问另一个位于 Woodridge的公司客户站点(不妨把它称为公司B),这个公司主要生产面包以及各类甜食。在我工作的第一天,公司的操作管理员提出希望有一个能够在(包括周末在内的)所有时间都可以正常工作的数据库系统(也就是 24×7可用性系统)。当时,我很困惑地想,一个面包生产公司的职员怎么会在星期六的晚上去访问本公司的Oracle数据和系统呢?

同时,我利用UNIX内核命令设计了一个简短的程序代码,用来周期性地访问这个公司的数据库并记录数据库使用情况。我将这个程序运行了整整一周(包括周末),并且每10分钟执行一次。表1-1中列出了我想要获取的信息。

然后,我带着这个报告找到了操作管理员,并且告诉他数据库系统使用的高峰期是正常的工作时间,本公司并不需要 24×7数据库系统。而且,如果试图使用 24×7系统,反而会损失数据管理灵活性。无庸置疑,我的这个建议同样没有得到重视。这样,我只能回到自己的办公室,等待下一个差事。

表1-1 公司B中的数据库每天的使用情况

时 间 段	使 用 情 况
08:00到18:00	平均60个并发用户会话
18:00到20:00	平均15个并发用户会话
20:00到22:00	平均6个并发用户会话
22:00到00:00	平均2个并发用户会话
00:00到04:00	没有用户登录(正在进行热备份)
04:00到07:00	没有用户登录(完成热备份;没有任何活动)
07:00到08:00	平均10个并发用户会话

通过上述两个例子,可以得出这样的结论:公司 A应该使用24×7正常工作时间数据库系统;而公司B则错误地认为自己需要 24×7正常工作时间的系统并且试图建立这种系统。下面分析运行24×7数据库系统所需要的开销。

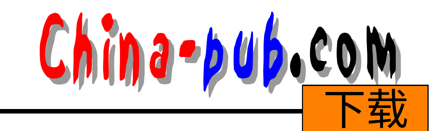
1. 24 ×7正常工作时间的开销估计

在24×7正常工作时间中,主要有以下因素影响其开销。

(1) 管理灵活性

对于新手,一个被普遍认为非常简单的任务有可能几乎不可解决。这些任务包括:

- 对数据库或操作系统打补丁
- 冷备份
- 重新组织表空间和表
- 创建/重建索引



• 利用VALIDATE STRUCTURE选项分析表/索引

实际上，对于任何需要对重要的表进行上锁的任务，数据和系统的重启或者机器的重新启动都是不允许的。顾客绝对不允许在数据库访问的高峰期出现机器崩溃的现象（特别是在机器长期没有被重新启动的情况下更是如此）。笔者曾经发现许多个滥用机器以及数据库的站点，它们存在各种各样的性能问题。所有这些问题都有可能导致系统的重新启动，从而重新组织那些成为碎片的内存。也就是说，为了获取高可用性所损失的第一个要素就是管理的灵活性。

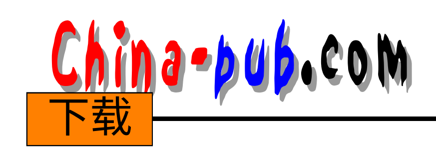
注意 在这里笔者并不赞成频繁地或者是毫无原因地重新启动机器或数据库。事实上，笔者只是推荐保证机器在大多数情况下都是在不会造成重大性能下降的条件下运行。但在某些情况下，总会存在一些导致内存分块或者内存泄漏的硬件 bug或软件bug（操作系统、Oracle或者其他软件）。例如，内存泄漏，即内存一直被某个程序所霸占，而没有被释放（这常常表现为使用了malloc()来获取内存，而没有使用相应的free()来释放内存，或者内存指针错误）。这时，用户就可能会需要花费大量的时间来分析并查找出现这些问题的原因，然后使用适当的工具（硬件、操作系统、Oracle支持）或者同时使用多个工具来修复这些问题。多数时候，这些修复工作并不是立刻就可以进行的，而是必须等待开发商提供相关的补丁之后才能解决。这样，按照预先定义的间隔时间（例如每四个月或者六个月一次）重新启动机器就变得非常必要（以便快速地释放浪费的内存以及被分块的内存）。对于进行重新启动的决策，必须在对内存分块情况经过非常细致的评估之后进行。

那么，是不是在24×7方式下运行Oracle就是一项非常艰辛的任务呢？关于这个问题，并没有非常简单的答案。如果对所使用的Oracle系统进行了非常详细的需求分析以及体系结构设计，那么Oracle系统就会在用户预先定义的可用性以及性能需求下正常工作。而如果用户试图将一些非常昂贵的硬件简单地堆积在一起来组织一个满足自己需求的数据库系统，这是不可能的。这样做所遇到的第一个问题就是性能问题，另外还有系统可用性问题。

(2) 性能

在现实世界中，性能常常（并不总是）与数据库的正常运行时间成反比（如图 1-1所示）。RAID 0就是这种反比关系的一个典型例子。在RAID 0中，为了保证系统性能，数据被通过多个磁盘/磁盘控制器进行了划分。但是，每个磁盘都成为单一失败点（在RAID 1或者其他映像机制不可用的情况下）。这时，所涉及的磁盘数目越多，失败的概率就越大。为了在数据库中演示一个关于本问题的例子，可以减少数据库的检查点（就是将所修改的数据块缓冲区写入到磁盘的数据文件上），从而可以通过减少物理I/O的方法来提高性能。相反，过少的检查点也会导致系统停工时间的延长，这是因为在数据库崩溃之后需要有很长时间来执行系统恢复过程。

关于数据库的另一个例子是在UNRECOVERABLE或者NOLOGGING模式下创建索引来加速数据库建立过程，以便提高系统性能。但是，如果数据库在索引被建立之后就崩溃了，这种情况也会增加数据库的停工时间。数据库恢复过程不能恢复最新所创建的索引，从而必须在数据库可用之前手工地创建这个索引，因而会延长停工时间（在后面的段落中将会详细描述UNRECOVERABLE/NOLOGGING操作是如何导致更长的停工时间的）。当然，也会出现例外的情况。例如，一个非常迅速的索引创建过程（特别是对于具有上兆或者更多行的表来



说）就有可能会使这个表很快对于所有的应用程序都可以访问（只要加在 CREATE INDEX语句上的锁被释放）。因此，从这个角度看，利用UNRECOVERABLE模式或者NOLOGGING模式实际上是增强了数据库系统索引的可用性（除非在索引创建之后立刻就发生了数据库崩溃的现象）。

注意　系统性能与可用性并没有天然的反比关系。理论上来说，可用性是性能的一个子集（如果系统没有任何吞吐量，那么它就不会有任何可用性）。但从实际的观点看，性能与可用性是两个相互独立的现象。性能通常与"数据库运行速度有多快"相关，而可用性通常与"用户在什么时候需要访问数据库"相关。笔者提供了几个例子用来说明它们两者（UNRECOVERABLE操作与CHECKPOINT操作）之间所存在的反相关性。

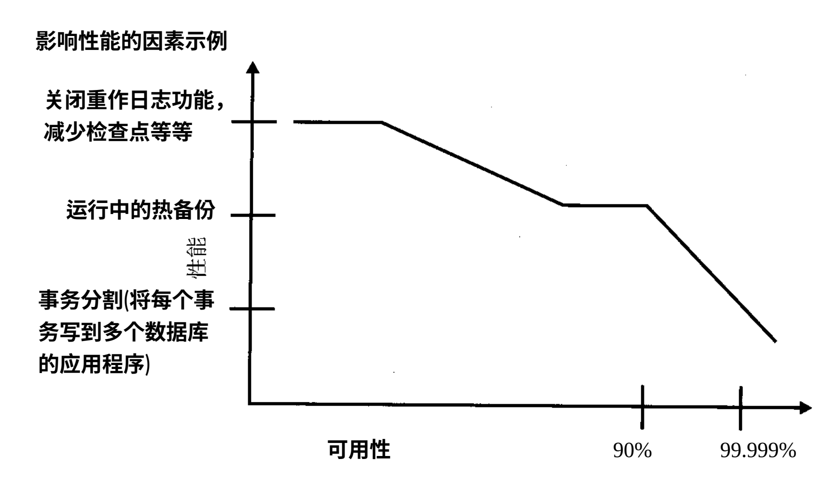


图1-1　用来说明在真实世界中数据库性能与可用性之间反比关系的图表

让我们绕一个弯来说明UNRECOVERABLE/NOLOGGING操作是如何导致更多停工时间的。这个例子可以适用于Oracle8i之前的系统环境。一个创建索引的操作会导致表被锁定，从而阻止并发进行的DML（插入、更新与删除）操作。因此，在Oracle8i之前的版本中，如果在一个频繁被访问的表中有一个索引正在被创建，那么这个表目前就是不可用的。如果这个表是一个被频繁访问的表，那么所有那些访问这个表的应用程序都必须被中断。而如果没有这些应用程序，那么这个数据库对于终端用户就是不可使用的。下面让我们来看一下表1-2和表1-3中的两个数据库脚本。

表1-2　UNRECOVERABLE/NOLOGGING模式下表创建过程的事件

脚本1

事 件 序 列	所导致的停工时间
UNRECOVERABLE/NOLOGGING模式下创建索引	30分钟
数据库崩溃	
从上一天晚上的备份中恢复数据库	30分钟
数据库开启并且可以使用	
所创建的新索引丢失	
索引被重新创建	30分钟
总停工时间	90分钟

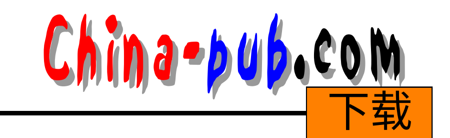


表1-3 RECOVERABLE/LOGGING模式下表创建过程的事件

脚本 2

事件序列	所导致的停工时间
RECOVERABLE/LOGGING模式下创建索引	40分钟
数据库崩溃	
从上一天晚上的备份中恢复数据库	30分钟
数据库开启并且可以使用	
所创建的新索引有效	
总停工时间	70分钟

以上两个脚本说明虽然UNRECOVERABLE可以帮助改善数据库性能，但导致总的停工时间增加；而且在第二个数据库脚本中，索引是在 RECOVERABLE模式下被创建的，虽然索引创建时间较长，但在数据库崩溃之后总的停工时间反而降低了。

需要说明的是，虽然由于采用24×7正常工作时间有可能会降低系统的性能，同时也有可能会降低确保这种性能可被接受的管理灵活性，但是事情并没有这么简单。在保证系统能够工作于24×7方式的条件下，用户没有必要违心地接受（或者强迫用户 /管理员接受）由于24×7方式所带来的低性能，或者认为系统的高性能与高可用性是不可共存的。在现实中，用户必须在进行系统安装的过程中就同时考虑到系统的性能与可用性，并且将这两者作为进行系统配置时的目标，而不能只兼顾其中一方。毕竟，一个在性能上非常低劣的数据库系统所提供的吞吐率与响应时间都是不能接受的。在实际应用中，如果这种性能过于低下并达到某个临界点，那么就可以认为这个系统不能使用。在这里笔者要说明的一个问题是，虽然在性能与可用性之间存在一定的对立关系，但用户必须采取一种合适的手段使得这两者都能够最大限度地发挥效能，而不必为了迎合某一个指标而减弱另一个指标，这一点对于 DBA、系统分析员、数据创建人员以及体系结构设计者都非常重要。关于这些方法，将在后续章节中详细描述。例如，可以在进行适当的分析、减少表空间中的冗余段、减少频繁的维护动作之后进行STORAGE参数的指定。所有的这些步骤都不是十分容易理解，如果用户需要为自己的公司实现一个24×7方式的数据库系统，必须对于数据库领域中的各种信息（包括数据库内部的知识与数据库外部的知识）都非常熟悉。此外，数据库系统中的许多监视工具都是强制进行的，这些监视工具能够进行数据库中的页管理以及监视数据库管理人员在使用数据库时可能产生的错误，从而有效地防止数据库因人为使用而导致的出错可能。关于这一点，即使是对于具有很多管理人员的公司也是非常必要的。只有这样，才能缩小数据库可用性与性能之间的差异，从而有效地防止这两者之间的差异不断扩大。

(3) 经费开销

除了数据库系统的管理灵活性与性能之外，笔者将要讨论的另一个问题就是开销问题。实现一个具有高可用性的系统其代价可能是非常昂贵的。在开销与可用性之间存在一个非常直接的关系，这就是：希望获取的可用性越高，那么开销也就越大（如图 1-2所示）。例如，假设贵公司的数据库系统在每周工作日的上午9点到下午5点之间可以被访问，并且由于计算机系统或者数据库系统的原因而有时会导致数据库不可被访问。如果在这种情况下，贵公司的生产效益不会造成过大损失，那么公司的数据库操作人员就可以仅仅配置单独的一台机器（无需使用备份机器）以及仅仅执行常规的备份来尽量地满足公司的系统可用性需求。虽然有些时候公司数据库系统的失败会在正常工作的时间发生，但公司对于这种情况是可以容

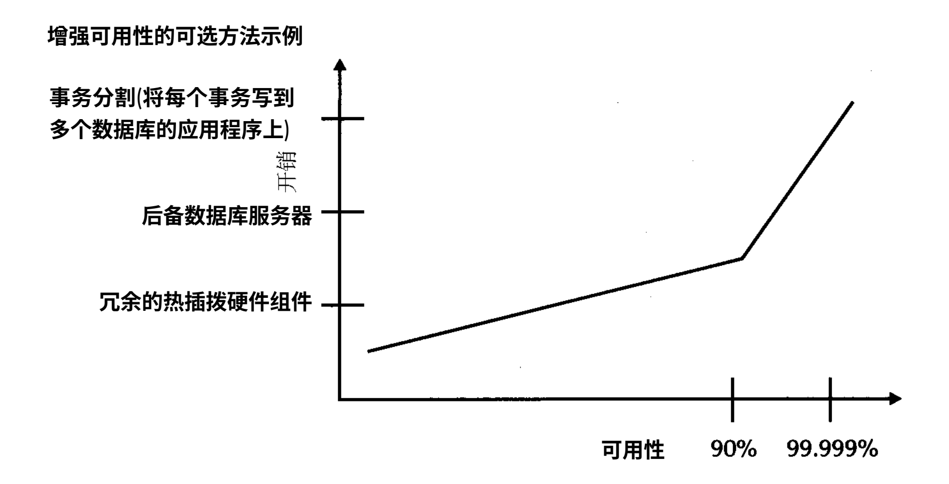


图1-2　数据库系统开销与可用性之间的直接关系

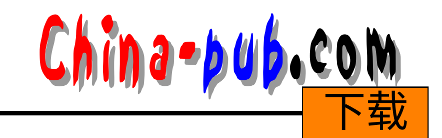
忍的，因此，公司就没有必要在 IT预算方面考虑使用冗余的硬件或者系统支持，从而节省了开销。但是，如果贵公司的数据库可用性需求比这个更高，例如需要保证数据库的 24×7可用性或者8×5可用性（也就是必须保证每天八小时、每周五天工作日中必须有效），而且公司决不容忍在这些时间段内数据库系统的失效，这时，就应该考虑使用非标准化系统构建技术（包括允许使用冗余硬件、自动恢复系统、更多的管理人员），从而导致IT开销的增加。总之，用户所希望的可用性越高，那么所需要的开销也就越高，因为必须为实现这种高可用性进行硬件/软件/系统维护方面的附加投资。

2. 评价系统可用性需求

介绍完关于24×7正常工作时间系统在性能、价格以及灵活性之间的相互关系之后，现在来分析一下系统可用性需求的评价方法。贵公司真的需要自己的数据库系统每周七天、每天24小时的全天候可访问功能吗？或者，是否贵公司只是需要自己的数据库系统每周五天、每天24小时的可访问性？或者，贵公司是否只是需要自己的数据库系统每周七天，每天 18小时的可访问性？如何确定本公司的数据库系统可访问需求呢？人们总是习惯于要求自己的数据库系统具有全天候的可访问性，似乎在任何时候，总是有人在自己的数据库中进行有关的访问动作。

关于这个问题最有效的解决方法就是直接与用户交谈。有时读者也许会惊奇地发现，用户对于自己所提出的问题是如此的热心，特别是在告知用户必须花费几百万美元才能实现真正的24×7数据库可访问模式（因为需要在多个地点使用多个数据库服务器并且利用高速网络连接到所有的这些服务器上提供快速复制）时，尤其如此。 (需要说明的是，在笔者提到复制(replication) 时，并不只是 Oracle中的Advanced Replication可选项——在提及Oracle中的Advanced Replication可选项时，使用的是字母“ R ”开头；而是使用“ replication ”来指代各种各样的复制工具，其中包括 Oracle中的Advanced Replication以及在市场上可以购买的第三方硬件或者软件复制工具。)

如果公司数据库系统的用户并不是那么合作，那么另外一个评价这些需求的方法就是收集关于数据库使用的统计信息（这时读者将会发现笔者在前面所提到的工具非常有用，关于



这个工具的详细说明请参见第19章），然后利用这些统计信息进行分析以支持自己的观点。

一般来说，每周星期六晚上以及星期天早晨是维护数据库系统的最佳时机，这是因为人们在这个时间段内对数据库的访问最少（如果有的话）。即使在上面所说的时间段内有一部分人访问数据库，这也并不意味着就不能在这个时间段内进行数据库维护。也许，这些用户可以被请求在别的时间段进行数据库访问。同时，如果在这个时间段安排了自动批处理任务，那么用户的访问请求就可以被安排在这个时间段之后进行（也就是维护工作完成之后），或者，这些用户的访问请求可以安排在周末的晚上运行。但是，如果确实因为现实的商业原因或者经济上的原因使得这些数据库访问请求不能被推迟，那么贵公司也许就确实需要运行于24×7方式的数据库系统。

在这里，笔者并不是打消贵公司使用24×7方式数据库系统的积极性，相反，笔者的目标是确信公司确实需要具有高可用性的数据库系统，并且确保贵公司做好了为实现24×7方式而花费的代价（这种代价包括购买必要的硬件资源以及由于数据库管理灵活性的损失而造成的额外开销）的准备。此外，对于公司数据库系统可用性需求的分析能够有效地判断数据库系统所存在的弱点，包括：

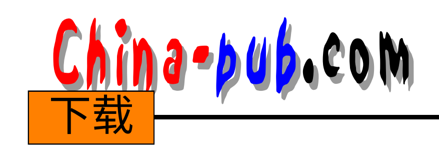
- 在什么时间公司的数据库系统必须完全可访问？应用程序访问数据库系统的高峰期与低谷期是什么？是否存在对数据库系统访问的高峰周/月（例如每年年终记帐时间），在这种高峰期数据库的事务比平常时间要多？
- 数据库系统访问的高峰期与低谷期之间的潜在差异有多大？
- 数据库是否能够部分可访问（也就是说，是否允许在某个时间内某些应用程序不能访问数据库或者某些表空间处于离线状态）？若是，这些应用程序与表空间是什么？
- 数据库系统各个组件之间是否存在功能相关性与物理相关性？
- 如果数据库系统中的某个特定组件不可使用，其他各个应用程序将会作出什么反应？
- 如果数据库系统的主服务器正在进行维护动作，应用程序是否仍然能够在一个慢速的、功能较弱的后备服务器上继续运行？
- 上述这种降低了的数据库访问性能对于贵公司是否可以接受？

在进行数据库正常访问时间需求分析之前，必须考虑与公司实际相关的一些关键问题。所有这些问题的目标就是理解本公司对于数据库系统停止工作的容忍度。可以说，对于数据库系统工作负载情况的特点研究越深入，那么就能够越准确地了解到数据库系统的正常工作时间需求，从而能够最大限度地节省开销。在实际的商业活动中，决定自己的数据库系统是否需要使用24×7方式是非常重要的，如果数据库系统的停止工作会对公司造成非常巨大的损失，那么使用24×7方式就非常必要。

3. 评估数据库系统不可访问情况下所造成的“实际损失”

由于数据库系统不可访问所造成的损失可以使用直接损失与生产力损失来衡量。关于这些损失，可以使用不同的标准来进行评价，例如：

- 在数据库系统停工期间，所造成的顾客/销售额损失（其中包括公众影响以及不利于本公司的报导的损失）。
- 在数据库系统停工期间，由于资源的空闲所造成的损失（“隐性”损失）。
- 在数据库系统停工期间，为了使得系统能够尽快恢复，聘请系统修复人员（例如，聘请数据库系统恢复专家）所花费的开销损失。如果公司投资使用了冗余的硬件/软件来保



证数据库系统的24×7方式，那么公司对于数据库系统停工的忍耐力就会更强，因为在主数据库系统停工的情况下，还可以使用备份硬件来完成相应的任务。相应地，在许多情况下，虽然利用公司内部的数据库系统恢复进行数据库恢复会花费更长的时间，但由于已经有了能够正常工作的备份系统，这种时间是可以容忍的，从而节省了聘请数据库系统恢复专家的开销。这样，在主数据库系统停工需要进行恢复的情况下，仍然可以保证数据库系统的正常访问。但是，如果所使用的是普通的系统（就是没有失败恢复功能的系统），那么如果数据库系统在数据库访问的高峰期停工，这时公司就必须考虑聘请有经验的数据库恢复专家进行处理，从而有可能导致开销的增加。

• 对于某些应用领域（医药、航空/交通控制等领域），有时数据库系统的不可访问有可能会导致人员的伤亡。

所有上面列出的这些方面，都必须在实际的损失分析之前进行考虑。例如，在考虑各种因素之后，如果公司的损失为$5 000（最坏的情况下），那么，公司是否值得使用$1000 000来保证数据库系统的连续可用性呢？此外，由于来自许多应用程序的数据（例如订单输入信息(Order Entry)、物料收据(Bill of Materials)、记帐信息(Accounting)等等）最终都要汇集到一个数据库中，那么就会很有必要规定所有这些应用程序的访问优先权。如果对所有的应用程序都规定相同的访问优先级，那么数据库系统的恢复过程就可能会非常复杂。在实际的工作中，必须保证"如果本应用程序的访问被推迟，公司的商业活动仍然能够正常进行"。例如，在订单输入信息中，如果公司不能及时地获取到客户的订单，那么就不仅会失去本次订单，而且有可能永久性地失去这个客户。这种情况如果经常发生，那么公司的销售底线就有可能被突破，从而导致公司利润的降低。因此，只有有效地控制公司数据库系统的正常工作时间，才能保证公司的正常生产效率。

面对这个问题，存在的一个障碍就是数据库管理人员必须对各公司以及技术部分有深入的了解，而且即使是对任何一部分缺乏认识都有可能导致错误的决策。随着数据库管理人员对数据库成长过程以及数据库使用模式的认识不断增强，那么对数据库可用性与性能的管理与预测将会变得非常容易。这也正是为什么24×7方式的数据库系统在初始时难以实现的原因之一。

1.1.2　理解"数据库"中的各个组件

"数据库"系统是由什么部分组成的呢？或者说，是什么实际地构成了数据库系统呢？数据库系统是否由物理文件与内存结构的组合构成？这是笔者对DBA进行面试时经常提出的一个问题。关于这个问题的回答，笔者所得到的答案也是千奇百怪的。从技术角度来看，对这个问题的许多回答都是正确的。但是，笔者个人倾向于以下的回答：

任何数据库都是依赖于许多层次的，而且每个层次都与其他层次同样重要，如图1-3所示。这些层次之间可能是相互交迭的，而且对于DBA是透明的。但无论如何，这些层次可以概括为：

• 硬件层
• 操作系统层
• 网络层
• DBMS（数据库管理系统，在这里就是Oracle）层

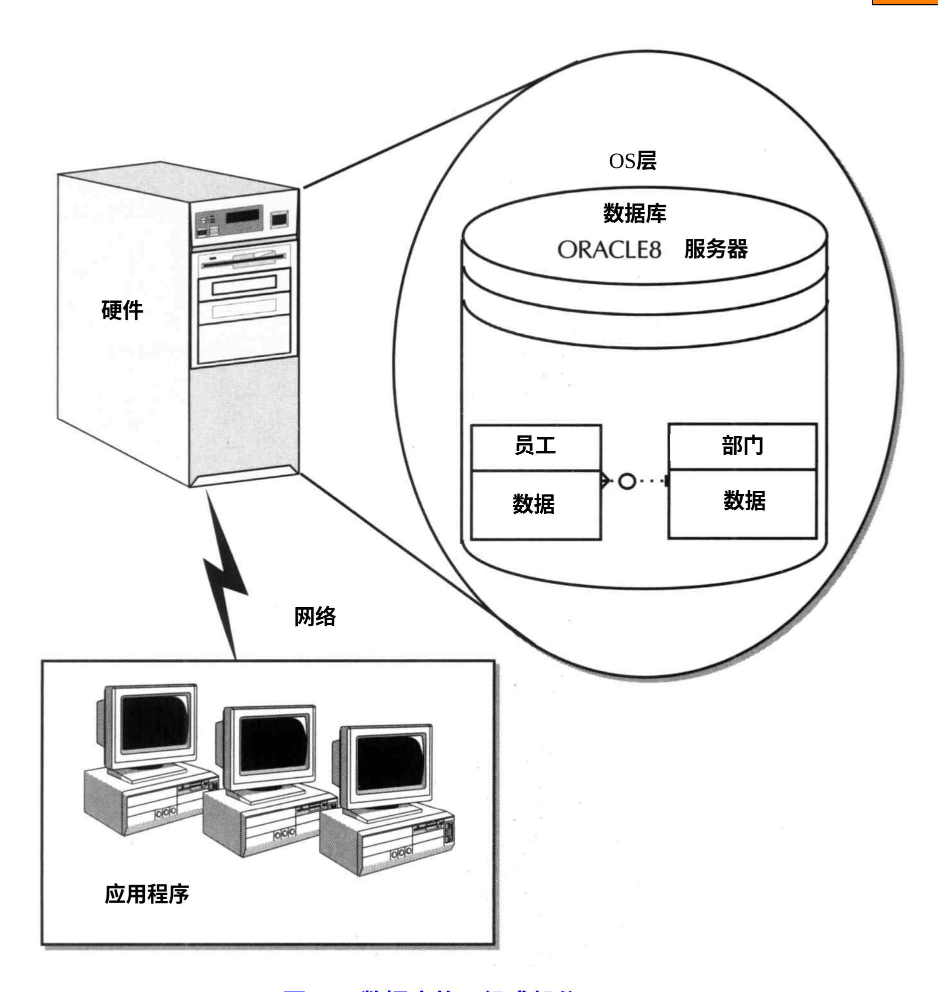


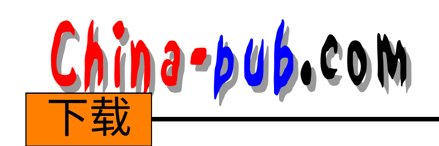
图1-3 数据库的“组成部分”

• 使用DBMS的应用程序层
• DBMS所管理的数据层

可以说，任何层都不能孤立存在。即使是 Oracle公司的“ Raw Iron ”，其中也在某种程度上有操作系统的风格。只要上述层次中的任何一部分被破坏，公司的数据库系统就不可访问，这也是24×7方式的一个不足之处。

对于本章中的每个问题，由于数据库系统的各部分之间是互相依赖的，它们都必须被单独地进行合理的配置与性能优化。任何一个称职的 DBA都知道Oracle的缺省配置并不是最有效的。Oracle非常复杂，因此在进行实际的安装与配置之前，必须进行合理的规划与自定义，以便使系统适合于公司特定的环境。现在许多 Oracle工具也具有非常友好的用户界面（特别是在NT 系统下），它们能够允许一个经验不太丰富的 DBA直接安装Oracle系统。但是，如果要保证Oracle系统的工作效率非常高，并不是一件容易的事情。这是因为 Oracle系统中的每个层次都必须进行合理的配置，才能保证系统整体工作效率的提高。

Oracle系统安装过程中的每个组件都必须单独地进行考虑，并且对每个组件的配置都必须像对待DBMS的配置一样进行。可以说，一个 Oracle系统的成功安装，需要用户对于许多方面的知识都非常熟悉。



一个非常称职的DBA必须具有数据库管理、网络管理、系统管理、应用程序开发以及数据管理方面的知识。在现实世界中，是否真正存在这样一个完全称职的DBA呢？回答是“不存在”。因此，另外一个解决方案就是把对上述各方面比较熟悉的DBA集合起来构成一个DBA小组，其中的某些组员熟悉系统管理，其他的组员则熟悉网络管理、应用程序开发以及数据管理等等。根据应用环境的实际需要，公司有时甚至会需要Web管理员来构成一个完整的“24×7工作小组”(特别是在公司利用网络实现电子商务应用时更是如此)。

实现公司数据库系统的高可用性是一项非常精确的工作，而不是一项含糊的工作。只有当数据库系统中的各个组件都能够正常工作时，才能够使数据库系统达到真正的可用性。下面将分析数据库系统中的各个组件对于数据库整体可用性的影响：

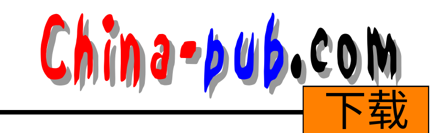
- 硬件 Oracle数据文件所存放的磁盘。如果硬件崩溃，所导致的结果是：数据库系统停工而不可访问。
- 操作系统 操作系统需要更多的交换区。操作系统目前到达最大的可用交换区，使得系统不能再进行内存分配。从而，系统反复地进行调页、交换(swap，或称对换)、页中断而最终导致系统崩溃。操作系统出错所导致的结果是：数据库系统停工而不可访问。
- 网络 网络通信电缆被剪断。网络出错所导致的结果是：顾客不能通过网络连接到公司的数据库系统上，从而使得数据库系统不可访问。
- DBMS 由于对某个经常被访问的数据库表需要进行重建与索引，导致需要在这个表的重建过程中将表锁定，使得所有使用这个表的应用程序都必须被中断。DBMS出错所导致的结果是：数据库系统不可访问。
- 应用程序 在应用程序的使用过程中发现了一个bug，因此必须使这个应用程序停止工作进行修复。这时，在后续的两个小时内，用户将不可访问这个应用程序。应用程序出错所导致的结果是：数据库对于那些使用这个应用程序访问本数据库的用户不可访问。
- 数据 假设公司的数据库系统中有一个由22个表所构成的数据存储单元。源端系统操作方的数据库数据批处理工作不能正常进行，并且数据库系统数据产生方的系统出现故障，那么这时数据就会出错。数据出错所导致的结果是：数据库系统不可访问。

使得数据库系统工作于24×7方式并不是需要将DBMS调节到能够达到最好的数据库性能方式，这个过程包括对数据库中所有组件的深入了解以及仔细调试。这意味着必须对数据库组件中各种错误的预测。对数据库系统中每个组件的所有可能导致出错的地方都必须严格检查，并且能够非常迅速地制定出一个实用、可行的方案快速恢复所导致的错误。例如，对于数据库系统中的关键硬件部分，如果这个硬件部分出错，能够有冗余的组件立刻对它进行替换(并且这种替换是在没有任何中断的情况下完成的)。

关于数据库各个组件的问题本章就讨论到这里，至于详细内容，读者可以参见相关的参考手册。但是，为了保证本书内容上的完整性，笔者将从Oracle系统的角度对这些组件作简要的说明。因此，本书的后续章节将分别讨论关于数据库系统硬件、操作系统、网络、应用程序以及数据管理方面的问题。

1.1.3 分析数据库系统中容易出错的部分以及出错时所产生的症状

如果一个数据库系统没有进行任何适当的维护，而连续运行了几个月，那么数据库系统



将有可能出现什么问题呢？回答很简单：会有许多错误！下面将列出可能发生事件类型（这些事件通常在深更半夜发生）。当然，这里列出的事件仅仅是作为一个例子，在破坏性程度上也许并不是非常详尽的（毕竟，每天晚上都有可能发生一些新情况）。

1. 人为的假信号（用户错误或者管理错误）

这一类型的错误通常比其他错误更为常见。人为错误通常可以分为两类：用户错误与管理错误。用户错误常常表现为用户具有过多的权限，从而可以访问没有被授权的一些数据。如果贵公司的数据库系统曾经有过被未授权的用户“偶然地”访问过的经历，那么就很容易理解笔者的意思。另一方面，管理错误常常是由于贵公司的计算机系统的网络或者数据库管理人员过度劳累或者其他错误所导致。如果读者曾经见过某个CIO（首席顾问）对一个新聘请的系统管理员进行呵斥，骂他在管理网络的过程中所带来的错误而导致服务的丢失时，就会理解笔者的意思。

需要说明的是，这一种错误也可以归纳为“缺乏安全性”。发生以上错误的最终结果就有可能使新聘请的DBA被认为非常懒惰，从而被解雇。

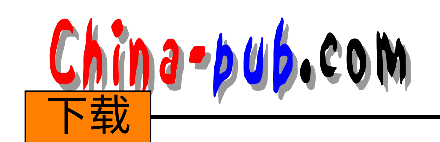
2. 硬件/软件出错

正如前面所述，数据库系统中的每个组件要么是由可能会出现bug的软件构成的（如操作系统、数据库），要么是由可能会导致失败的硬件构成。无论在哪种情况下，软件bug或者硬件失败都有可能会导致整个系统的崩溃。无论何时，只要出现了软件bug，就必须使用适当的补丁来进行修复，从而最终需要重新启动计算机系统。事实上，即使是出现软件bug，许多操作系统（例如UNIX和NT）也必须由于内存/页面错误、核转储、内存交换、内存出错等等的原因而重新启动。许多普通的系统管理员或者DBA都习惯于在这种情况下直接进行系统重新启动。但是，在一个高可用性环境中，这种做法是不可思议的。

在购买硬件之前，还有必要进行充分的需求评估。对于许多高可用性站点，他们在购买新硬件之前都会利用自己所开发的测试程序利用真实的数据对此硬件进行测试。数据库系统操作人员必须对每个硬件的技术指标有充分的了解，并且对硬件的理论MTBF（Mean Time Between Failure，失败平均间隔时间；详情请见后面关于MTBF的讨论）与实际MTBF非常熟悉。例如，如果某个硬盘驱动器的理论MTBF（这是导致硬件失败的常见原因）为500 000小时，这就表示从理论上来说，硬盘驱动器可以连续地在24×7方式下工作57年。同时，许多公司也利用非常真实的模拟环境去对产品进行测试。它们首先测试的是环境的改变（包括操作系统补丁、操作系统发布以及新环境）。并且只有在那些新的改变经过充分的认证之后，才能将它们加入到数据库系统应用中。所有为建立这些测试环境所花费的开销都应该算作为了获取数据库系统高可用性而花费的开销中的一部分。

3. 环境失败

读者也许曾经有过接收到住房管理部门关于在星期六将要断电的通知（这种情况一年中也许总会发生几次）。碰到这种情况，用户也不用过于担心，这是因为后备的电源仍然可以保证使用至少六个小时。除了电源方面的问题之外，其他环境因素包括工人罢工等等也会影响数据库系统正常工作。几年前，当笔者还是一个化工厂的全职DBA时，工厂内部的工人举行了一次罢工，那时笔者无法进入工厂内部。这样，笔者就不能启动数据库系统并且实现远程控制。甚至是连进行数据库系统的备份动作也不行，因为根本就不能对磁带驱动器中的磁带进行更换。



4. 自然灾害与人为灾害

这些情况包括地震、水灾、火灾、战争、暴乱等等。

这些事情在任何时候都有可能会发生，关键问题在于：是否为此作好了准备？如果明天就发生这些情况，该怎么办？是否需要为所有这些情况都进行防范，或者仅仅需要防范软件、硬件以及用户错误等问题？这些也是笔者常常问客户的一些问题。可以说，在平时需要 24×7正常工作时间的客户，在战争或者地震时也许就不需要这种方式。当然，存在许多跨国公司，他们要求即使是出现了这些情况也要实现数据库系统的 24×7方式，特别是对于电子商务更是如此。但是，如果贵公司的规模并不那么庞大，虽然也许会要求对这些突然事件进行防护，但没有必要为此支付高额的代价来实现 24×7方式（在这些突发事件下）。那么，什么才是最佳的解决方案？必须在实现某个解决方案之前花费充分的时间来理解公司的商业目标，然后再拟出一个解决方案的草案。如果这个解决方案与公司的商业目标相一致，那么就有可能得到一个非常成功而且容易被接受的解决方案。

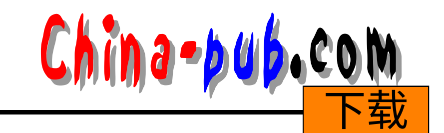
1.2　实现一个“理想的”24×7系统

本节将讨论24×7数据库系统的特点并且给出适当的步骤来实现这些特点。也许其中的一些特点比另外一些特点会更加适合于贵公司的实际需要，但是，在进行实际解决方案的设计过程中必须将这些因素都考虑在里面。

1.2.1　24×7数据库系统的共同目标

一般来说，理想的24×7数据库系统在体系结构上应该具有以下特点：

- 健壮性　这个数据库系统应该能够自动处理许多系统失败以及数据库失败。并且必须对每个单一失败点进行标识，同时为每个这种组件提供了相应的冗余组件。
- 透明的失败恢复　系统应该提供最大限度的失败恢复透明性，使得主数据库系统在崩溃时能够尽量不影响系统的整体可用性。需要说明的是，失败恢复不仅是指数据库系统失败的检测功能，同时还包括作出相应响应动作的过程。必须提供一种机制，能够周期性地检测数据库系统的稳定性。如果将这种周期性检查的周期时间设置过短，那么就有可能会由于经常性的检测动作而导致系统整体性能的下降；而如果将这种周期性检查的周期时间设置过长，那么就有可能导致数据库系统服务的失败被终端用户所感觉到。此外，一旦主数据库系统的失败被检测到，所采用的失败恢复策略必须能够重新发现数据库的新连接路由，以便尽快地与数据库系统建立连接。这时需要注意的是，先前的数据库连接有可能是活跃的也有可能是非活跃的。其中对于数据库连接的活跃性判断非常关键，24×7方式必须保证数据库连接中的应用程序具有重新启动功能。如果这种功能不能实现，那么数据库的解决方案中就必须提供一种机制，能够在必要的情况下重新建立应用程序与数据库系统的连接，从而减少应用程序反复调用数据库的开销。有些时候，数据库系统服务的中断很容易被数据库访问用户所发现，这是因为用户可以明显地感觉到数据库系统连接被中断。
- 数据完整性　24×7方式下的数据库系统必须保证数据的完整性，不能允许出现数据的丢失或者数据内部相关性的冲突。例如，在将所有的数据从主数据库系统传送到备份数



据库系统的过程中，不允许出现任何数据丢失的现象。

- 经济承受能力　实现数据库系统的开销以及数据库系统自身的开销，应该低于不采用24×7方式时的比例。在短时期内，24×7方式数据库系统的开销也许会比较高（只要打开这段时间以来的记帐本就可以发现）。但是，随着时间的推移，这种高开销就会明显地下降，这是因为24×7所带来的效益。关于这一点，只有在将24×7方式下数据库系统的开销与潜在的系统停工时间所带来的效益损失进行比较才可以发现。在进行损失计算时，只要考虑在数据库系统访问的高峰期这个特殊情况就可以得到最坏情况下的损失大小。因为，在其他时间，所造成的损失应该不会大于此时的损失。
- 优化资源使用　24×7方式下的数据库系统应该以尽可能优化的方式来使用系统资源，以便不会到达系统的饱和状态（特别是在数据库系统访问的高峰期）。
- 系统实现的简单性　将公司的数据库系统移植到24×7方式下的工作量不应该过大，并且系统停工时间以及服务质量的下降（对于用户来说）必须保证在用户可接受的范围之内（这个范围应该作为考虑系统方式移植的一个因素）。同时，对相关组件（包括应用程序、安全策略等等）所作的修改必须具有一定的可行性。
- 系统可维护性　必须保证公司内的系统管理人员能够在不求助于不可行方式（例如，频繁的脱产培训、复杂的维护工具、高级专家顾问的常规性访问）的情况下进行系统维护。
- 性能　在日常的工作中，新的数据库系统不应该影响系统性能。需要说明的是，某种程度上的性能影响也许会是不可避免的。例如，如果24×7方式要求所有的应用程序都必须通过多个数据库（包括位于远程站点的数据库）才能正常工作，那么网络延迟就有可能会影响应用程序的正常工作。但是，必须保证这种影响在可接受的范围之内。

也许，最终所得到的解决方案并不能成功地达到上面列出的所有目标（特别是经济承受能力方面）。但是，一个理想的解决方案应该是兼顾尽可能多的目标，并且重点考虑对于公司来说最重要的目标。本书后续章节中所讨论的提示信息可以帮助从数据库的角度以及系统整体的角度进行实际可操作解决方案的选择。本书第2章将讨论前面所列出的各种情况下所采用的具体解决方案类型。

1.2.2　利用服务级协议管理系统需求

SLA（Service Level Agreement，服务级协议）是一种正式定义关于系统可用性与系统性能的特定需求问题的文档。只有经过终端用户与系统管理者之间的协商讨论之后，才有可能达到需求上的一致性。SLA的目标就是确保系统管理者与终端用户在数据库服务质量上的一致性。其中详细地说明了对于系统失败的定义以及从这种失败中进行恢复所需要做的工作。一个优秀的SLA也会指定性能与可用性之间的折衷关系以及为了实现系统高可用性与高性能所花费的代价之间的关系。公司内部的SLA同时也是公司与第三方硬件/软件提供者之间建立外部SLA的依据。

笔者曾经看到过许多公司的数据中心都没有非常合理的SLA支持。而即使是那些拥有SLA的公司，它们也不会对于SLA中所列出的条款给予足够的重视。但是，对于那些寻求数据库系统高可用性的公司来说，SLA中的每个条款都是非常重要的。除非对于自己的需求非常明确；否则，如果不满足SLA中所列出的条款，就无法实现数据库系统高可用性。

可以说，公司终端用户的期望是决定数据库系统正常工作时间需求的重要依据。也许，贵公司需要同时支持不同类型的、具有不同需求的终端用户（例如，数据表表项操作人员、中级管理者与高级管理者等等）。其中有的终端用户只在他们的工作时间需要访问数据库；而另外一些终端用户（例如VP与CEO）每天只需要访问几分钟，但在这短短的几分钟之内，他们需要运行资源密集型的图形报表或者相似的应用程序。因此，可以说终端用户的期望是非常模糊并且多变的。必须判定什么用户要求对于公司是非常重要的而另外一些需求是不重要的，并且保证这些判断对于今后公司的发展确实有帮助。因此，必须以 SLA的形式在公司（操作管理部门）与特定的终端用户之间进行详细的说明。不过如果公司的数据库终端用户在地理位置上是分散的并且不是十分容易确定（例如在电子商务中），那么SLA就必须基于高级规范来列出基本的数据库可用性需求。这种高级规范可以来自对数据库使用模式的实时跟踪，或者来自一般的普遍规律。例如，一个正常的金融应用站点都满足这样一个规律：数据库系统必须在每天的上午8点到下午6点之间保证可访问。因此，最初的 SLA就必须满足这样一个准则。然后，通过对数据库系统的使用情况进行一定时间（例如两个月，或者更长的时间）的实时跟踪之后，就可以得到关于数据库可访问性的更准确需求情况，从而可以制定新的SLA规范。

一般来说，SLA必须满足以下特点。

1. 完整性

一个优秀的SLA必须列出所有影响公司业务的关键因素。例如，为了保证公司的生存，数据库系统的停工时间不能超过三个小时，这个问题就必须在 SLA中明确说明。如果公司的数据库系统需要必须保证能够允许至少200个并行会话同时进行，也必须在SLA中说明。

2. 必须使用真实的数据

公司数据库系统的所有需求问题都必须用真实的数据进行量化。也就是说，在 SLA中必须使用精确数据来描述每个用户需求问题。例如，如果公司的数据库系统需要跨地区使用一个备用数据库，需要花费的开销为X；而由于使用了这个备用数据库，对于主数据库系统所带来的性能压力为Y，那么这样两个问题就必须在SLA中进行说明。当然，X与Y也可以不是真实的开销数据，但必须是真实的百分比数据。这样，就可以帮助终端用户与系统管理人员在系统高可用性问题上进行折衷处理。如果单纯地说明使用这个数据库所带来的性能减弱，并不能非常有效地解决问题。因为有了适当的真实数据就可以增加自己的说服力，从而使得它更为可信，而不是建立在模糊的假设之上。在这里，SLA又可以看做是一种合同协议。因此，在SLA中不能简单地说“如果A，那么将会B”，而应该更为清晰地指出“如果A，那么将会B；而如果C，那么将会D。而且，后一种方案的代价是前一种方案的两倍”。

3. 适应性

SLA必须具有动态可适应性，并且能够反映最新情况。许多公司在建立 SLA之后，就会在几个月之后完全忘记其中的内容。造成这种问题的原因之一，就是 SLA的固定性，以致于SLA不能适用于实际的情况。大多数情况下，在 SLA的创建过程中公司会考虑到当时的实际情况，但是随着公司的不断发展壮大，这些情况也会作相应的改变，而不是静止不变的。因此，必须对SLA作周期性的修改，以便能够适应公司具体情况的改变。特别是在网络时代，用户对数据库访问的模式改变得非常迅速，更加应该保证 SLA的可扩充性以及兼容性。如果根据公司的实际情况，要求每两个月就必须重新制定 SLA，那么公司也必须进行相应的SLA



制定。例如，在电子商务的初期，公司中的所有顾客都只能算是“ Guest ”，那么这时的SLA就必须基于这样一个考虑来制定。随着电子商务的发展，那么就有一批顾客会经常使用公司的数据库系统，那么公司就完全有必要重新修改自己的 SLA。可以说，某个公司的 SLA版本越老，那么它与公司目前实际情况之间的差异就越大。在公司决定购买新的硬件或者软件时，就必须根据SLA中的条款作出决策。这时，就非常有必要立刻更新 SLA。

4. 内容清晰

在SLA中所定义的条款必须非常精确与清晰。必须对其中的每个条款给予足够的解释，使得不能对其中的任何条款产生二义性解释。例如，如果只是在 SLA中说明“要求采用高可用性系统”，而不对其作任何解释，那么这种条款是没有意义的。而且，应该在条款中指出必须保证的正常工作时间，同时列出对系统进行维护时间声明。如果同一个数据库需要为多个应用程序服务，那么在 SLA中可以给出所有这些应用程序的累积需求，或者为每个应用程序分别给出其可用性需求并为所有的应用程序产生一个综合的高级可用性需求表。此外，如果能够以每天中的时间分段的方式给出系统可用性需求，那么对于公司的数据库系统可用性开发是非常有意义的。表 1-4提供了一个说明系统正常工作时间需求的文档示例。此外，还必须说明系统的性能需求。例如，应该在文档中指出，如果系统中有 150个用户，那么小型的报表可以在10分钟或者更短的时间内给出结果；但如果系统中的用户超过了 150个（例如在高峰期），那么这种报表就需要 30分钟才能给出结果。因此，必须在 SLA中列出高峰期性能与非高峰期性能、关键应用程序性能与非关键应用程序性能。

表1-4 根据数据库系统的应用程序使用情况进行正常工作时间需求分析

时间段	正常工作时间需求	在此时间段内所运行的应用程序
08:00～18:00	非常重要	物料管理、销售、资金管理、产品规划、生产管理
18:00～23:00	重要	物料管理、生产管理
23:00～04:00	未被使用	
04:00～08:00	重要	生产管理、数据仓库、批处理工作

根据上面的需求分析，操作管理部门就可以决定何时进行系统维护（例如备份与索引重建）。同时，将所有运行在特定时间段上的应用程序进行罗列，有利于 DBA决定什么类型的部分可访问性对于终端用户是合适的（也就是说，什么类型的表空间可以在某个时间内被脱机，而什么类型的表空间又不能被脱机）。在数据库系统恢复的过程中，这些信息可以帮助 DBA决定哪一个应用程序需要首先被恢复。如果数据仓库批处理工作没有必要立刻运行，那么就不必首先执行数据仓库表的恢复动作；相反，可以将销售应用程序表优先恢复。

5. 经济可行性与技术可行性

一个有效的SLA必须考虑公司经济可行性与技术可行性的问题。在 SLA中所列出的任何高可用性解决方案都不应该仅仅是在技术上可能的，而且必须考虑本公司的实际承受能力。任何不能同时满足这两个要求的 SLA条款都是不可行的，因此也不适于实际的实现。例如，在确保24×7系统可用性的情况下，如果需要不损失任何性能来提供实时的系统备份，那么可以说对于贵公司的实际环境是从技术上不可接受的。但是，如果在另外一个数据中心（越洋）进行全部数据库系统的备份是可能的，那么可以说，这种方案对于贵公司是从经济上不可接受的，因此也是不可取的。这样，为了能够保证 SLA成功地被实现，必须由公司操作管理部

门代表、公司高级管理层代表以及终端用户代表三者之间进行协商来决定。终端用户应该提供最基本的系统需求信息，公司操作管理部门应该提供可能的解决方案，而公司的高级管理部门应该主要从经济上来考虑解决方案的可行性问题。对于操作管理部门所给出的在技术上可行的方案，在公司高级管理部门看来有可能不可接受（主要是由于开销原因）。如果由于为了实现系统的高可用性而投入的开销高于这种高可用性所带来的效益，那么确保系统的高可用性的行为就变得没有意义（至少是对于高投资的那部分是如此）。某些时候，高可用性的解决方案也许会来自终端用户。例如，如果在白天运行某些类型的会话密集型批处理工作代价过高，那么终端用户也许就可以不运行这种应用程序或者在晚上运行这种应用程序。在这种情况下，这种解决方案就来自终端用户。公司的操作管理人员一般并不知道是否某些批处理工作可以被推辞。

6. 可接受的平均时间

一个有效的SLA必须直接指定（或者利用附加文档指定）对于数据库系统可接受的 MTBF时间或者MTTR时间。MTBF（失败之间的平均时间）是一个系统在失效之前平均的工作时间。例如，如果某个数据库系统在由于磁盘失效或者网络失效而导致崩溃之前，正常运行了 600小时，那么就可以说此系统的 MTBF为600小时。典型情况下，数据库系统的 MTBF与实现此数据库系统的组件的MTBF直接相关。

有两种类型的MTBF：理论MTBF与操作MTBF。理论MTBF是由工具开发者在受控的实验环境下对被测试系统进行高强度测试的情况下所得到的统计值。理论 MTBF测试中所得到的数据并不能真实地反映现实生活中的实际情况（例如人为管理错误）。操作MTBF是在现实的生活中所得到的关于组件的MTBF值。

为了帮助读者更好地理解 MTBF，下面给出一个示例。 ABC公司有一个运行于 Sun Enterprise 4000上的Oracle8数据库系统，其中的数据分布于 EMC 3100磁盘阵列上（利用RAID-S技术）。假设这个磁盘阵列的MTBF为250 000小时，并且使磁盘阵列由32个磁盘构成。那么，这个磁盘阵列上每个磁盘的MTBF平均值为7 800小时（250 00小时/32个磁盘＝7 800小时/磁盘）。也就是说，大约每 10个月就有一个磁盘会发生故障（ 7 800小时/24小时每天/30天每月＝10月）。因此，ABC公司中数据库系统的MTBF不会超过10个月（这是因为公司数据库系统的正常工作时间依赖于所使用的磁盘 MTBF）。此外，在求取数据库系统的真正 MTBF时，还必须考虑其他的关键组件（例如，网络与包括CPU与控制器等等在内的硬件设备）的MTBF。一般情况下，数据库系统的总体MTBF永远不会超过其中所有关键组件 MTBF的最小值。

MTTR（Mean Time to Recover，平均恢复时间）是数据库系统在失败之后，进行系统恢复所需要花费的时间。仍然可以使用上面的例子来帮助理解 MTTR这个概念。假设这个Oracle8数据库系统在每天晚上执行联机数据库备份。并且进一步假设备份是在早上 2点到早上4点之间进行。进一步假设这个数据库系统有四个联机重新执行日志文件备份的动作，并且每个小时有两个日志文件会进行更新（因此每小时会产生两个新的归档日志文件）。现在假设在数据库恢复过程中，传送每个归档的日志文件需要花费 3分钟。那么，如果数据库系统在数据库被访问的高峰期（上午 11点）由于磁盘控制器的原因而失效，那么就大约需要花费 8个小时才能将所有的数据库操作进行恢复，如表 1-5所示。在这种情况下，数据库系统 MTTR就是8小时。

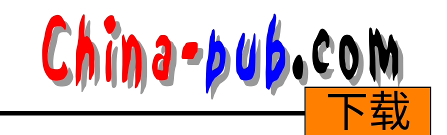


表1-5 从磁盘控制器失效中恢复过来的必要任务

任 务	所需的时间
更换损坏的磁盘控制器	6小时（这个时间的长短取决于磁盘供应商所提供的磁盘性能）
从磁带上将前天晚上所备份的信息恢复到磁盘中	30分钟
执行数据库恢复动作（在这个过程中总共有14个已经归档的日志文件需要被恢复：7小时×2个日志文件/小时＝14日志文件）	42分钟（平均每个归档日志文件3分钟）
实行数据库系统恢复动作（恢复未确认的数据）	15分钟
重新开启数据库系统以及所有的应用程序	8分钟
缓冲建立时间（为了对先前执行任务进行恢复）	30分钟
总计所需的时间	8小时（大约）

对于MTBF的最简单理解方式就是数据库系统的平均正常工作时间；相似地，对于 MTTR的最简单理解方式就是数据库系统的平均失效时间。

通过前面的讨论，读者可以知道 SLA的制定实际上是对某些问题进行研究、评估以及测试之后所得到的结果。为了能够实现 SLA中所提到的目标，必须非常合理地对公司中的各项资源进行配置。只有在对数据库系统中所有的薄弱环节进行完整的分析之后，公司才有可能给出足够多的解决方案来解决本公司的数据库系统正常工作时间需求问题。

1.2.3 只有90%的系统可用性就意味着节省90%的系统开销

在进行本公司的数据库系统正常工作时间需求分析时，需要考虑到的一个问题就是“ 90%规则”。这个规则就是，一个90%可用的数据库系统，可以节省公司中90%的开销。也就是说，数据库系统可用性需求中的其余 10%占用了实现这个数据库系统可用性中 90%的开销（为了达到24×7 方式的要求）。对于数据库系统正常工作时间需求的了解是非常重要的，表 1-6给出了正常工作时间需求的百分比与真实世界时间之间的关系。正如前面所说的，一个 90%可用性的系统，意味着每天必须容忍平均 2.4小时的系统停工时间，或者说每 10天有1天的系统停工时间，或者说每年有 36.5天的系统停工时间。通常，如果公司内部有对于数据库操作比较熟练的职员并且他们能够执行一般的数据库系统维护与监视动作，那么公司的数据库系统仍然会偶尔失败。有时，硬件或者软件的失败还需要更换一个组件或者使用软件补丁。同时，环境灾难的影响（例如，台风或者地震）也会导致数据库系统的破坏，当然这些因素的影响取决于数据中心所在的位置。如果这些突发事件发生的可能性非常小，那么公司就可以考虑不对它们采取任何补救措施。因此，公司必须能够准确评估这种事件发生的频率以及造成损失的大小。在这里，笔者想要说明的一个问题就是：在考虑这些突发事件因素的情况下，公司是否值得花费大量的开销来避免这些突发事件造成的损失？根据笔者的观察，数据中心只需要进行常规的数据备份与适当的数据恢复规划，就可以保证数据中心的数据信息具有较高的可用性。也就是说，系统中所提供的标准系统创建技术就足以为数据库系统提供 90%的可用性。

表1-6　真实世界的正常工作时间统计

正常工作时间百分比	每年的停工时间
90.0	35天，12小时
99.0	3天，15.5小时
99.9	9小时
99.99	50分钟
99.999	5分钟

通过对后续章节的学习，读者将能够实现组件失效与数据库维护任务的性能价格比相对较好的高可用性解决方案。但是，这种方案中并不会考虑到自然灾难或者人为灾害的影响。这是因为如果需要保证在出现这些灾难的情况下数据库系统仍然有效，就必须在另外一个地理上不同的位置对数据中心的数据进行备份。经过对公司商业需求、自然环境、所有的数据库组件进行深入的分析之后，必然能够得出正确的正常工作时间需求结论。

1.2.4　理解系统维护动作对于可用性的影响

系统操作人员经常提到的一个术语就是“维护”。那么，为什么在每个周末的晚上需要进行数据库系统的维护呢？为什么每周六的深夜需要将数据库系统的电源关闭呢？其回答很简单：因为需要进行维护。所有的操作管理人员与系统管理人员都意识到了系统 /数据库维护的重要性。

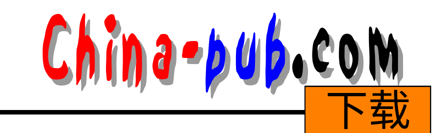
为了保证任何系统的平滑工作，必须对它进行动态的调整。在 24×7方式下，系统维护任务可以分为以下几类：

• 前摄维护
• 响应维护

前摄维护（proactive maintenance）是基于实际的数据库监视经验以及某种被认为有效的维护经验而进行系统维护工作。关于前者的一个例子就是在一个快速成长的数据库系统上，根据某些区段（extent，或称扩展段）的使用情况来决定为了避免“MAXEXTENTS reached”出错而增加系统对出错的健壮性。关于后者的典型例子是在不同的磁盘集合上放置索引段，而将这些索引段的相应表段放置在另外一个位置，从而基于 Oracle中的最优灵活性体系结构（Optimal Flexible Architecture，OFA）避免这两者之间的冲突。

响应维护通常基于数据库系统所遇到的最后数据库监视错误或者某种特定的错误来进行维护。关于前一种方式的例子是在紧急情况下将另一个数据文件添加到 DATA表空间中（例如在表空间中只有3MB的空闲空间，而下一个需要使用的区段却需要 5MB大小的空间）。关于后一种方式的例子是在每个应用程序用户都在自己的计算机屏幕上出现“ORA-1540：Unable to allocate 5242880 bytes in tablespace (DATA不能在表空间DATA上分配5242880字节）”消息之后，CIO要求将另一个数据文件添加到DATA表空间上。

由于需要具有高度的前摄维护支持，使得系统高可用性需求常常会导致 catch-22情况的出现。许多时候，进行这种维护需要更长的系统停工时间，从而又导致高可用性最终目标的降低。因此，必须寻求一种革新的解决方案，能够在不影响系统可用性的情况下按照一定的间隔时间进行有计划的前摄维护操作（第 15章到第18章将讨论这种类型的解决方案）。最后需要说明的是，在数据库系统或者其他任何新的应用程序运行的初期，进行系统维护的频率应该



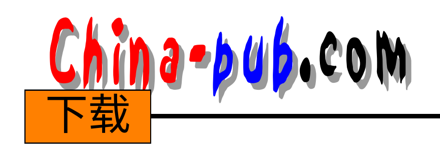
较高，这是因为在这种情况下用户使用模式尚未确定。

对系统所需进行的所有维护动作都必须在 SLA中定义并列举出来，并且所有相关的技术部门都必须对这些维护操作有足够的认识，从而能够更好地了解到系统的正常工作时间需求（包括：这种需求是什么？为什么需要这种需求？这种维护需求应该多长时间执行一次？）。数据库系统维护人员完全有必要了解什么操作有可能会导致系统的停工以及导致系统停工的原因；是否有其他的方法能够在不导致系统停工的情况下实现同样的维护动作？表 1-7中列出了一些关于维护操作的示例。当然，读者可以根据自己公司的设计需求对这个表进行适当的修改，使得它能够作为本公司 SLA的一部分。

注意 表1-7中所列出的“频率”并不能作为指导读者完成某项工作的实际频率，这里它只是用来说明某项任务多长时间应该执行一次。在实际应用中，某项工作的执行频率是与多种因素相关的，并且每个站点之间都各不相同。

表1-7 给出了执行频率与停工时间需求的实例动作任务

任 务	频 率	停工时间需求
重建表（磁盘碎片整理、行链重建）	每个季度执行一次，或者在进行非常重要的数据加载之后执行一次（而不论时间间隔的大小）	Yes
重建索引（磁盘碎片整理）	每个季度执行一次，或者在进行非常重要的数据加载之后执行一次（而不论时间间隔的大小）	Yes
为基于开销的优化器分析段	每天晚上执行一次	No
为进行结构合法化检查分析段（用来检查数据库退化）	每个季度执行一次（在出现系统将要崩溃的迹象时，立刻执行）	Yes
备份（热备份）	每天晚上执行一次	No
备份（对预先确定的表/模式进行输出）	每个星期执行一次	No
运行统计应用工具（例如 utlbstat、utlestat以及其他自行开发的应用工具）	每个星期执行一次	No
修改段属性（存储参数、FREELISTS、INITRANS、PCTUSED、PCTFREE以及并行度等等）	根据实际的需要执行	对于某些操作
修改静态数据库初始化参数（init.ora参数）	根据实际的需要执行	Yes
创建新的表空间，向已有的表空间中添加新的数据文件	根据实际的系统容量计划执行（事实上也是根据实际需要执行）	No
将数据库系统升级到一个新版本，执行必要的补丁操作	根据实际的需要执行	Yes
将某种特定的数据拷贝到另一个数据库系统中（从可操作的数据源加载到数据仓库与数据中心）	每天晚上执行一次	No
将已归档的数据库重作日志转移到脱机介质上	每天晚上执行一次	No
改变联机重作日志文件的大小，创建更多的重作日志文件	根据实际的需要执行	No
重建整个数据库系统（改变数据库块大小、将数据库转移到另一个台计算机上、对整个数据库进行磁盘碎片整理）	根据实际的需要执行	Yes
将已归档的重作日志信息拷贝 /应用于备份数据库服务器	根据实际的需要执行	No



1.3 半专业技术管理员与主管人技巧

下面给出了一些对于半专业技术管理员与主管人十分有用的技巧。

1.3.1 24×7方式需要对操作管理实施的基本方法

24×7方式并不仅仅是一种系统需求，而是一种基本原理，它要求整个数据库系统管理小组（包括设计者与开发者在内）具有高级的前摄维护能力。如果整个数据库系统管理小组不能对于系统的高可用性需求有足够的认识，就有可能会造成小组工作上的松散。但是，在24×7方式下，这种工作的松散将会直接对数据库系统导致一些人为的灾难。公司数据库系统解决方案的优劣将会对数据库系统性能产生决定性的影响，一个好的解决方案可以使数据库系统提供连续在高峰性能下的运行，而一个差的解决方案将会使数据库系统经常停止工作。任何一种操作任务，不论它听起来有多么简单，管理员都必须对这种操作进行认真的评价，例如：这种操作会对整个系统与数据库造成什么影响？是否会对某个重要的表产生排斥性的锁定？这种操作的执行是否会导致数据库不可使用？这种操作是否需要许多资源的支持才能完成，并且是否会最终导致数据库系统运行速度的降低？

1.3.2 在关键数据库访问时间创建“24×7小组”

前面已经说过，仅仅一个DBA（或者DBA小组）并不能够解决数据库系统所出现的所有问题。每个DBA都应该有一个为其服务的支持/操作小组。一旦进入数据库访问的关键时期，就应该极力确保在每个时间帧都有足够的人员来应付数据库系统可能出现的各种问题。

理想情况下，每个24×7小组都应该由DBA、系统管理员与网络管理员组成，并且应该保证这些人员在相应的时间帧上在位。同时，公司的开发小组与数据管理员还必须为24×7小组提供服务支持，使得能够在关键时期辅助进行问题解决。下面就请看一个例子。

表1-8中列出了一个基于Web的数据库。对于某运行三个基于Web的数据库系统（系统总容量超过80GB）的公司，在表1-8中，列出了本公司数据库系统维护人员在各个时间帧上的工作安排情况。

表1-8给出了一种如何基于相应时间上的业务情况将完整的一天划分为不同的时间帧的方法。从表中可以看出，在数据库系统的所有“非常关键”时期，必须保证至少有两个DBA与系统管理员在位。当然，根据本公司业务需求的实际情况与经济承受能力，可以安排更多的人或者更少的人。

表1-8 基于时间帧的数据库系统人员需求

时 间 帧	时间帧对于公司业务的重要性	人 员 需 求
07:00～15:00	非常关键	主要人员（必须在位）： Susan M：DBA Marty G：DBA Joe A：系统管理员 Michelle P：系统管理员 Rob F：系统管理员 Dave S：网络管理员 Brain G：开发者



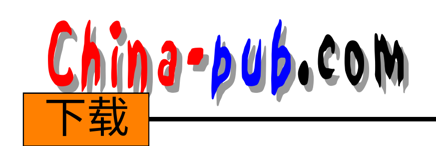
(续)

时 间 帧	时间帧对于公司业务的重要性	人 员 需 求
07:00~15:00	非常关键	Peter H：开发者/数据管理员 二级需求人员（随叫随到）： Sam J：DBA Bill K：系统管理员
15:00～19:00	非常关键	主要人员（必须在位）： Sam J：DBA Lucas S：DBA Thomas A：系统管理员 Vladimir P：系统管理员 Bill T：网络管理员 Patti N：开发者 二级需求人员（随叫随到）： Susan M：DBA Rob F：系统管理员
19:00～23:00	关键	主要人员（必须在位）： Sam J：DBA Lucas S：DBA Thomas A：系统管理员 Vladimir P：系统管理员 Patti N：开发者/数据管理员 二级需求人员（随叫随到）： Marty G：DBA Joe A：系统管理员
23:00～07:00	不关键	主要人员（必须在位）： John L：DBA Lisa W：系统管理员 二级需求人员（随叫随到）： Lucas S：DBA Vladimir P：系统管理员

但有一点需要说明的是，为这些潜在的数据库系统错误来安排解决人员并不能保证错误就不出现，这些24×7工作小组并不能完全保证阻止数据库或者系统出现任何错误。但是，安排这些人员之后可以达到以下两个目标：

• 对系统可能出现的错误进行预防，并采取相应的措施尽量阻止错误的出现。
• 在错误出现之后，能够保证有专门的人员来快速地恢复错误。

另一个需要考虑的问题是，必须了解数据库活动的高峰期以及数据库系统在某时不可用的情况下会造成多大的损失。如果能够确定在 19点到21点之间数据库系统出现问题会导致巨大的损失，那么就应该在这个时间段上确保有至少两个 DBA与系统管理员在位进行实时的系统修复与维护。但是，如果能够确定在这个时间段数据库系统出现问题所造成的损失并不严重，那么就可以考虑仅仅使用一个操作员或者仅仅在这个时间段安排一个 DBA与系统管理员



来监视系统。如果在这个时间段中出现了严重的情况，系统操作监视员就可以再请求更多DBA与系统管理员的支持。

表1-8中表明的另一个事实是，表中所列出的二级需求人员都是在完成了自己的主要工作之后进行安排的。因此，在进行人员安排时就必须注意安排的顺序问题，以便尽量不影响他们的个人休息时间。例如，如果 Susan M已经在7点到15点之间完成了自己的工作，那么如果再在20点又要求她回办公室进行工作，那么她将不会十分乐意（也许她正与自己的小孩在一起呢）。因此，在这个时间就应该安排别人作为二级需求人员，还应该将 Susan安排在另一个时间段作为二级需求人员。实践证明，这种考虑对于保持公司职员时间的协调关系非常重要，而这也是典型的24×7方式环境中必须考虑的一个重要问题。

此外，必须保证数据库系统维护人员不要同时休假。笔者认为，这是一个普遍的规则，几乎每个操作管理者与CIO都会意识到这个问题。但是，笔者曾经也经常听到以下这种对话：

OM（操作管理员）:“ Bob， 我们遇到了一个问题。数据库好像崩溃了。”

CIO :“ Jeff是否在位？ ”

OM :“ 哦，Jeff正在休假。”

CIO :“ 是吗！那么Lisa呢？ ”

OM :“ 很不幸，Lisa也在休假。它将在20号之后才能回来，…… ”

CIO :“ 好……，还有其他办法吗？是否能够找到 Jeff？ ”

OM :“ 可是目前我们并不知道 Jeff正在何处……，因为我已经在半小时之前找过他。 Joe是一个新手，我必须在这里监视系统的情况。好了，我过后将汇报更详细的情况…… ”

从这些事例中得出的一个结论是：必须确保在任何领域都有一个关键性人物存在，并且他知道其领域中的人员情况。

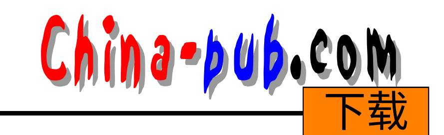
1.3.3 维护一个在数据库系统出现问题时需要告知的扩展人员列表

扩展人员列表是在数据库系统出现危机或者紧急情况时必须通知的人员。表 1-9中列出了扩展人员列表的一个例子。

表1-9 扩展人员列表示例

系统停工时间	需要通知的人员	被通知人员的身份
立刻	“ 随叫随到 ”维护人员	相关技术管理员（DBA、系统管理员、网络管理员）
0.5	Jeff A	操作管理员
0.5	Christine L	用户服务管理员
1.0	Sheila P	CIO
1.0	Greg S	用户服务VP
超过2.0	Timothy S	销售VP
超过2.0	Leslie G	CEO

这种扩展人员列表能够在系统停工期间更好地处理所出现的问题。例如，如果数据库系统已经停工，并且预计的停工时间会超过两个小时，那么就非常有必要告知销售 VP，以便能够让他（她）告知关键用户这些信息。这是因为，高级管理人员告诉关键用户本公司数据库系统目前所遇到的问题，远远优于让这些用户自己通过反复的登录企图失败而自己分析出本公司数据库系统所遇到的问题。



同样，当技术人员在了解到公司的CEO已经知道数据库系统停工并且某些技术人员正在恢复数据库系统时，技术人员解决这些问题的效率会更高（当然，必须保证CEO没有站在这些技术人员的后面监视他们）。在这里也存在一个平衡的问题，过早地将数据库系统出现问题告知其他相关人员会对DBA造成一定的压力，有时会使得他（她）的工作受到影响。

1.3.4 尽量在数据库系统停工之前预先通知客户或者终端用户

在笔者的经历中，终端用户与客户在自己发现某公司的数据库系统不可使用时都会变得焦躁不安，这不仅仅是因为数据库系统停工事件本身，而大多数是因为他们对事件的突然感到意外，因为这样打乱了他们的正常商业活动。如果能够在数据库系统停工之前预先通知他们，让他们对这种情况作出预先考虑，从而使得其所受到的影响降到最小。

如果数据库系统操作人员并不称职或者对终端用户不负责任，那么就极有可能会导致数据库系统突然停工；特别是在电子商务领域，如果出现这种情况就很难赢得顾客的信任，更别说与顾客建立一种密切的合作关系。

在数据库系统需要停工的情况下，最好的办法是在公司的主页或者Web站点上预先发出通知。若有可能，也可以向那些关键性的顾客发送电子邮件，将公司的这种情况告诉他们。理想情况下，根据数据库系统停工时间的不同，应该在24小时到48小时之内通知这些用户。

1.4 小结

本章重点讨论了实现高可用性系统之前必须考虑的第一个问题：精确地分析数据库正常工作时间需求。同时，本章还从技术角度与非技术角度阐述了24×7方式基本概念，列出了数据库系统中的主要组件，并且在最后给出了终端用户与操作管理员之间的一种协调工具：服务级协议（SLA）。

下一章将讨论数据库系统中各种各样的紧急事件，并且分析解决这些事件的方法。